EXHIBIT AB

STANDARDS REPORT

The H.264/MPEG4 Advanced Video Coding Standard and its Applications

Detlev Marpe and Thomas Wiegand, Heinrich Hertz Institute (HHI),
Gary J. Sullivan, Microsoft Corporation

ABSTRACT

H.264/MPEG4-AVC is the latest video coding standard of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). H.264/MPEG4-AVC has recently become the most widely accepted video coding standard since the deployment of MPEG2 at the dawn of digital television, and it may soon overtake MPEG2 in common use. It covers all common video applications ranging from mobile services and videoconferencing to IPTV, HDTV, and HD video storage. This article discusses the technology behind the new H.264/MPEG4-AVC standard, focusing on the main distinct features of its core coding technology and its first set of extensions, known as the fidelity range extensions (FREXt). In addition, this article also discusses the current status of adoption and deployment of the new standard in various application areas.

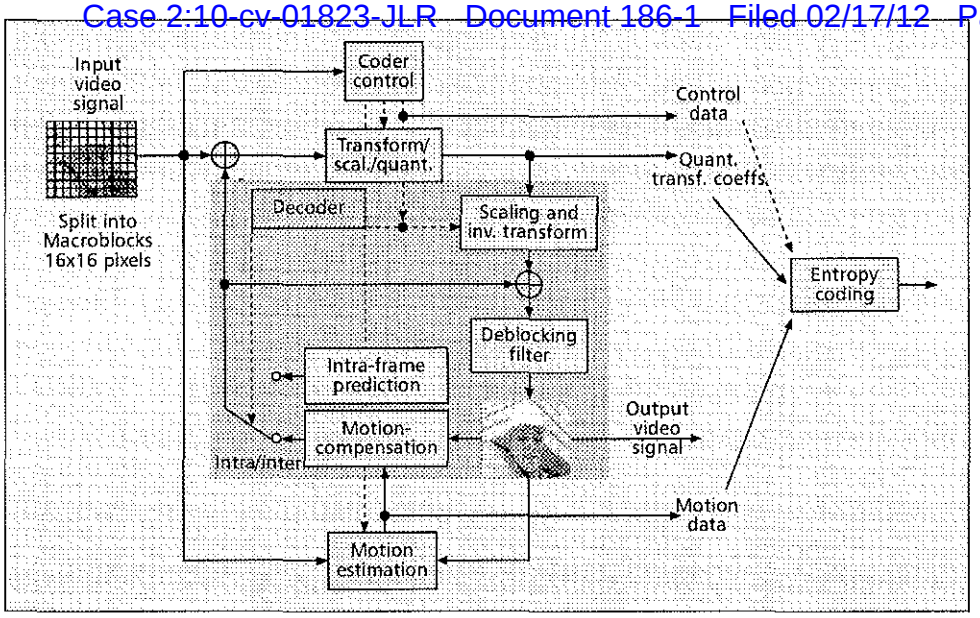
INTRODUCTION AND HISTORICAL PERSPECTIVE

Digital video technology is enabling and generating ever new applications with a broadening range of requirements regarding basic video characteristics such as spatiotemporal resolution, chroma format, and sample accuracy. Application areas today range from videoconferencing over mobile TV and broadcasting of standard-/high-definition TV content up to very-high-quality applications such as professional digital video recording or digital cinema/large-screen digital imagery. Prior video coding standards such as MPEG2/H.262 [1], H.263 [2], and MPEG4 Part 2 [3] are already established in parts of those application domains. But with the proliferation of digital video into new application spaces such as mobile TV or high-definition TV broadcasting, the requirements for efficient representation of video have increased up to operation points where previously standardized video coding technology can hardly keep pace. Furthermore, more cost-efficient solutions in terms of bit rate vs. end-to-end reproduction quality are increasingly sought in traditional application areas of digital video as well.

Regarding these challenges, H.264/MPEG4 Advanced Video Coding (AVC) [4], as the latest entry of international video coding standards, has demonstrated significantly improved coding efficiency, substantially enhanced error robustness, and increased flexibility and scope of applicability relative to its predecessors [5]. A recently added amendment to H.264/MPEG4-AVC, the so-called fidelity range extensions (FREXt) [6], further broaden the application domain of the new standard toward areas like professional contribution, distribution, or studio/post production. Another set of extensions for scalable video coding (SVC) is currently being designed [7, 8], aiming at a functionality that allows the reconstruction of video signals with lower spatiotemporal resolution or lower quality from parts of the coded video representation (i.e., from partial bitstreams). The SVC project is planned to be finalized in January 2007. Also, multi-view video coding (MVC) capability has been successfully demonstrated using H.264/MPEG4-AVC [9], requiring almost no change to the technical content of the standard.

Rather than providing a comprehensive overview that covers all technical aspects of the H.264/MPEG4-AVC design, this article focuses on a few representative features of its core coding technology. After presenting some information about target application areas and the current status of deployment of the new standard into those areas, this article provides a high-level overview of the so-called video coding layer (VCL) of H.264/MPEG4-AVC. Being designed for efficiently representing video content, the VCL is complemented by the network abstraction layer (NAL), which formats the VCL representation and provides header information in a manner appropriate for conveyance by a variety of transport layers or storage media. A representative selection of innovative features of the video coding layer in H.264/MPEG4-AVC is described in more detail by putting emphasis on some selected FREXt-specific coding tools. Profile and level definitions of H.264/MPEG4-AVC are briefly discussed and finally a rate-distortion (R-D) performance comparison between H.264/MPEG4-AVC and MPEG2 video coding technology is presented.

0163-6804/06/$20.00 © 2006 IEEE



■ **Figure 1.** *Typical structure of an H.264/MPEG4-AVC video encoder.*

The video coding layer of H.264/MPEG4-AVC is similar in spirit to that of other video coding standards such as MPEG2 Video. In fact, it uses a fairly traditional approach consisting of a hybrid of block-based temporal and spatial prediction in conjunction with block-based transform coding.

Applications and Current Status of Deployments

As a generic, all-purpose video coding standard that is able to cover a broad spectrum of requirements from mobile phone to digital cinema applications within a single specification, H.264/MPEG4-AVC has received a great deal of recent attention from industry. Besides the classical application areas of videoconferencing and broadcasting of TV content (satellite, cable, and terrestrial), the improved compression capability of H.264/MPEG4-AVC enables new services and thus opens new markets and opportunities for the industry. As an illustration of this development, consider the case of "mobile TV" for the reception of audio-visual content on cell phones or portable devices, presently on the verge of commercial deployment. Several such systems for mobile broadcasting are currently under consideration, e.g.,

- Digital Multimedia Broadcasting (DMB) in South Korea
- Digital Video Broadcasting — Handheld (DVB-H), mainly in Europe and the United States
- Multimedia Broadcast/Multicast Service (MBMS), as specified in Release 6 of 3GPP

For such mobile TV services, improved video compression performance, in conjunction with appropriate mechanisms for error robustness, is key — a fact that is well reflected by the use of H.264/MPEG4-AVC (using the version 1 Baseline profile described below) together with forward error correction schemes in all of those mobile-broadcasting systems.

Another area that has attracted a lot of near-term industry implementation interest is the transmission and storage of HD content. Some indications of that trend are shown by the recent inclusion of H.264/MPEG4-AVC (using version 3, i.e., FREXt-related "High profile" described below) in important application standards or industry consortia specifications such as:

- The revised implementation guideline TS 101 154 of the Digital Video Broadcasting (DVB) organization
- The HD-DVD specification of the DVD Forum
- The BD specification of the Blu-Ray Disc Association (BDA)
- The International Telecommunication Union — Radiocommunication Standardization Sector (ITU-R) standards BT.1737 for HDTV contribution, distribution, satellite news gathering, and transmission, and BT.1687 for large-screen digital imagery for presentation in a theatrical environment

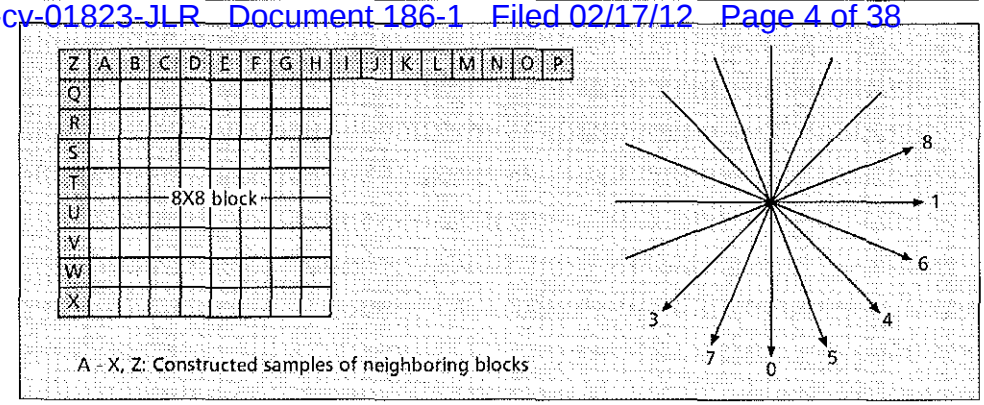
In addition, a number of providers of satellite television services (including DirecTV, BSkyB, Dish Network, Euro1080, Premiere, and ProSiebenSat.1) have recently announced or begun near-term deployments of H.264/MPEG4-AVC in so-called second generation HDTV delivery systems (often coupled with the new DVB-S2 satellite specification). At the time of writing this article, at least four single-chip solutions for HD decoding of H.264/MPEG4-AVC for set-top boxes are on the market by the semiconductor industry.

High-Level Overview of the Video Coding Layer

The video coding layer of H.264/MPEG4-AVC is similar in spirit to that of other video coding standards such as MPEG2 Video [1]. In fact, it uses a fairly traditional approach consisting of a hybrid of block-based temporal and spatial prediction in conjunction with block-based transform coding. Figure 1 shows an encoder block diagram for such a design.

A coded video sequence in H.264/MPEG4-

The typical encoding operation for a picture begins with splitting the picture into blocks of samples. The first picture of a sequence or a random access point is typically coded in Intra mode. For all remaining pictures of a sequence or between random access points, typically Inter coding is utilized.



■ **Figure 2.** *Samples used for 8 × 8 spatial luma intra prediction (left), and directions of 4 × 4 and 8 × 8 spatial luma intra prediction modes 0, 1, and 3–8 (right).*

AVC consists of a sequence of coded pictures [4, 5]. A coded picture can represent either an entire *frame* or a single *field*, as was also the case for MPEG2 video. Generally, a frame of video can be considered to contain two interleaved fields: a top field and a bottom field. If the two fields of a frame were captured at different time instants, the frame is referred to as an *interlaced-scan* frame; otherwise, it is referred to as a *progressive-scan* frame.

The typical encoding operation for a picture begins with splitting the picture into blocks of samples. The first picture of a sequence or a random access point is typically coded in *Intra* (intra-picture) mode (i.e., without using any other pictures as prediction references). Each sample of a block in such an Intra picture is predicted using spatially neighboring samples of previously coded blocks. The encoding process chooses which neighboring samples are to be used for Intra prediction and how these samples are to be combined to form a good prediction, and sends an indication of its selection to the decoder.

For all remaining pictures of a sequence or between random access points, typically *Inter* (inter-picture) coding is utilized. Inter coding employs interpicture temporal prediction (motion compensation) using other previously decoded pictures. The encoding process for temporal prediction includes choosing motion data that identifies the reference pictures and spatial displacement vectors that are applied to predict the samples of each block.

The *residual* of the prediction (either Intra or Inter), which is the difference between the original input samples and the predicted samples for the block, is transformed. The transform coefficients are then scaled and approximated using scalar quantization. The quantized transform coefficients are entropy coded and transmitted together with the entropy-coded prediction information for either Intra- or Inter-frame prediction.

The encoder contains a model of the decoding process (shown as the shaded part of the block diagram in Fig. 1) so that it can compute the same prediction values computed in the decoder for the prediction of subsequent blocks in the current picture or subsequent coded pictures. The decoder inverts the entropy coding processes, performs the prediction process as indicated by the encoder using the prediction type information and motion data. It also inverse-scales and inverse-transforms the quantized transform coefficients to form the approximated residual and adds this to the prediction. The result of that addition is then fed into a deblocking filter, which provides the decoded video as its output.

Main Innovative Features of the Video Coding Layer

This section contains a more detailed description of the main building blocks of the H.264/MPEG4-AVC video coding layer sketched in the last section. The innovative nature of the characteristic features of those coding tools can be best described as having a substantially higher degree of diversification, sophistication, and adaptability than their counterparts in prior video coding standards. After presenting the rather traditional concept of how pictures are partitioned into smaller coding units below, some of the most representative innovations of the H.264/MPEG4-AVC video coding layer are introduced step by step, largely following the order of processing in the encoder as described in the previous section.

Subdivision of a Picture into Macroblocks and Slices

Each picture of a video sequence is partitioned into fixed size *macroblocks* that each cover a rectangular picture area of 16 × 16 samples of the luma component and, in the case of video in 4:2:0 chroma sampling format, 8 × 8 samples of each of the two chroma components. All luma and chroma samples of a macroblock are either spatially or temporally predicted, and the resulting prediction residual is represented using transform coding. Each color component of the prediction residual is subdivided into blocks. Each block is transformed using an integer transform, and the transform coefficients are quantized and entropy coded.

The macroblocks are organized in *slices*, which represent regions of a given picture that

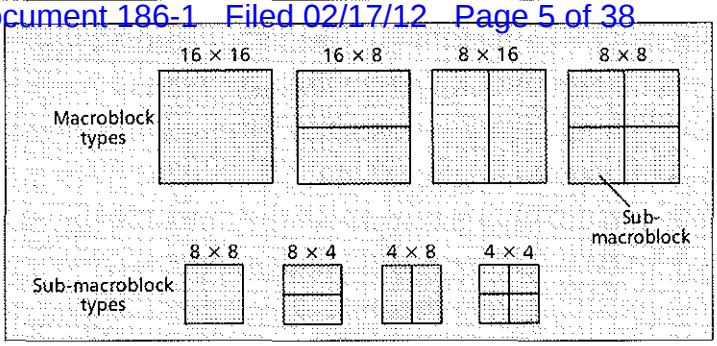
can be decoded independent of each other. H.264/MPEG4-AVC supports five slice-coding types. The simplest is the *I* slice (where *I* stands for intra). In *I* slices all macroblocks are coded without referring to any other pictures of the video sequence. Prior coded images can be used to form a prediction signal for macroblocks of the predictive-coded *P* and *B* slice types (where *P* stands for predictive and *B* stands for bi-predictive). The remaining two slice types are *SP* (switching *P*) and *SI* (switching *I*) slices, which are specified for efficient switching between bitstreams coded at various bit rates [5].

A *picture* comprises the set of slices representing a complete frame or field. Splitting an interlaced-scan picture to create separate pictures for each field is especially efficient for random access purposes if the first field is coded using *I* slices and the second field is predicted from it using motion compensation. Furthermore, field-based coding is often utilized when the scene shows strong motion, as this leads to a reduced degree of statistical dependency between adjacent sample rows (because the alternate rows of an interlaced frame are captured at different time instants). In some scenarios parts of the frame are more efficiently coded in field mode, while others are more efficiently coded in frame mode. Hence, H.264/MPEG4-AVC also supports macroblock-adaptive switching between frame and field coding (MBAFF). For that, pairs of vertically contiguous macroblocks in a coded frame are categorized as either two frame-segmented (i.e., vertically spatially neighboring) or two field-segmented (i.e., vertically interleaved) macroblocks. The prediction processes and prediction residual coding are then conducted using the selected segmentation.

Spatial Intra Prediction

Each macroblock can be transmitted in one of several coding types depending on the slice coding type. In all slice coding types, at least two intra macroblock coding types are supported. All intra coding types in H.264/MPEG4-AVC rely on prediction of samples in a given block conducted in the spatial domain, rather than in the transform domain as has been the case in previous video coding standards. The types are distinguished by their underlying luma prediction block sizes of 4 × 4, 8 × 8 (FREXt only), and 16 × 16, whereas the intra prediction process for chroma samples operates in an analogous fashion but always with a prediction block size equal to the block size of the entire macroblock's chroma arrays. In each of those intra coding types, and for both luma and chroma, spatially neighboring samples of a given block that have already been transmitted and decoded are used as a reference for spatial prediction of the given block's samples. The number of encoder-selectable prediction modes in each intra coding type is either four (for chroma and 16 × 16 luma blocks) or nine (for 4 × 4 and 8 × 8 luma blocks).

As illustrated in Fig. 2 for the case of 8 × 8 spatial luma prediction — a type that is only supported by FREXt related profiles — luma values of each sample in a given 8 × 8 block are predicted from the values of neighboring decoded samples. In addition, as a distinguished feature of the 8 × 8 intra-coding type, the reference samples are smoothed by applying a low-pass filter prior to performing the actual prediction step. Eight different prediction directions plus an additional averaging (so-called DC) prediction mode (corresponding to mode 2 and not shown in Fig. 2) can be selected by the encoder. The 4 × 4 and 16 × 16 intra prediction types operate in a conceptually similar fashion except that they use different block sizes and do not include the smoothing filter.

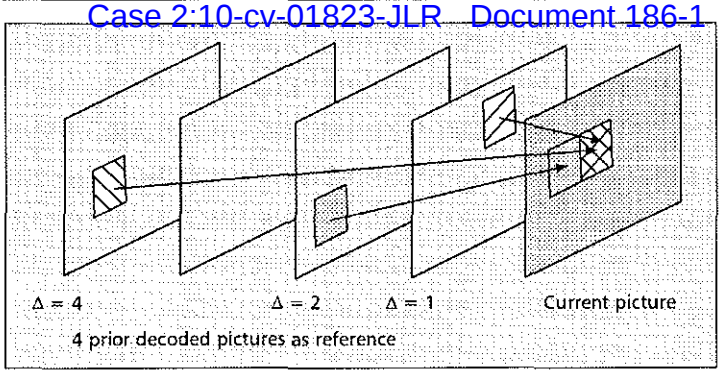


■ **Figure 3.** *Partitioning of a macroblock (top) and a sub-macroblock (bottom) for motion-compensated prediction.*

Motion-Compensated Prediction in *P* Slices

In addition to the intra macroblock coding types, various predictive or motion-compensated coding types are allowed in *P* slices. Each *P*-type macroblock is partitioned into fixed size blocks used for motion description. Partitionings with luma block sizes of 16 × 16, 16 × 8, 8 × 16, and 8 × 8 samples are supported by the syntax. When the macroblock is partitioned into four so-called sub-macroblocks each of size 8 × 8 luma samples, one additional syntax element is transmitted for each 8 × 8 sub-macroblock. This syntax element specifies whether the corresponding sub-macroblock is coded using motion-compensated prediction with luma block sizes of 8 × 8, 8 × 4, 4 × 8, or 4 × 4 samples. Figure 3 illustrates the partitioning.

The prediction signal for each predictive-coded $M \times N$ luma block is obtained by displacing a corresponding area of a previously decoded reference picture, where the displacement is specified by a translational motion vector and a picture reference index. Thus, if the macroblock is coded using four 8 × 8 sub-macroblocks, and each sub-macroblock is coded using four 4 × 4 luma blocks, a maximum of 16 motion vectors may be transmitted for a single *P*-slice macroblock. The motion vector precision is at the granularity of one quarter of the distance between luma samples. If the motion vector points to an integer-sample position, the prediction signal is formed by the corresponding samples of the reference picture; otherwise, the prediction signal is obtained using interpolation between integer-sample positions. The prediction values at half-sample positions are obtained by separable application of a one-dimensional six-tap finite impulse response (FIR) filter, and



■ **Figure 4.** *Multiframe motion compensation. In addition to the motion vector, picture reference parameters (Δ) are also transmitted.*

prediction values at quarter-sample positions are generated by averaging samples at integer- and half-sample positions. The prediction values for the chroma components are obtained by bilinear interpolation.

H.264/MPEG4-AVC supports multi-picture motion-compensated prediction in a manner similar to what was known as enhanced reference picture selection in H.263 v. 3 [3]. That is, more than one prior coded picture can be used as reference for motion-compensated prediction. Figure 4 illustrates the concept which is also extended to *B* pictures as described below.

For multi-frame motion-compensated prediction, the encoder stores decoded reference pictures in a multi-picture buffer. The decoder replicates the multi-picture buffer of the encoder according to the reference picture buffering type and memory management control operations (MMCO) specified in the bitstream. Unless the size of the multi-picture buffer is set to one picture, the index at which the reference picture is located inside the multi-picture buffer has to be signaled. The reference index parameter is transmitted for each motion-compensated 16 × 16, 16 × 8, or 8 × 16 macroblock partition or 8 × 8 sub-macroblock.

In addition to the macroblock modes described above, a *P*-slice macroblock can also be coded in the so-called skip mode. For this mode, neither a quantized prediction error signal nor a motion vector or reference index parameter are transmitted. The reconstructed signal is computed in a manner similar to the prediction of a macroblock with partition size 16 × 16 and fixed reference picture index equal to 0. In contrast to previous video coding standards, the motion vector used for reconstructing a skipped macroblock is inferred from motion properties of neighboring macroblocks rather than being inferred as zero (i.e., no motion).

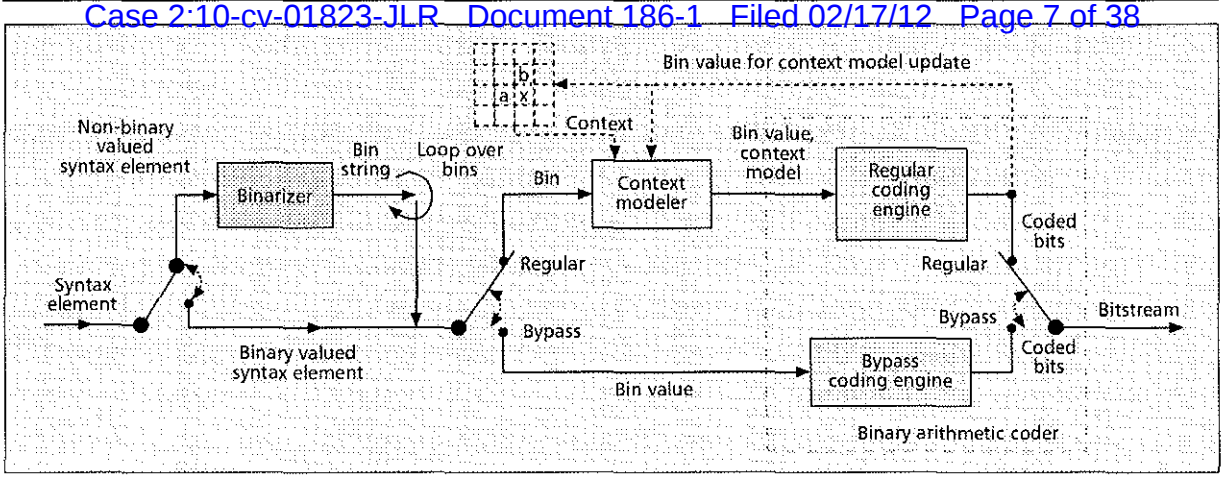
Motion-Compensated Prediction in *B* Slices

In comparison to prior video coding standards, the concept of *B* slices in H.264/MPEG4-AVC is generalized in several ways [5]. For example, unlike in MPEG2 video, B pictures themselves can be used as reference pictures for motion-compensated prediction. Thus, the only substantial difference between *B* and *P* slices in H.264/MPEG4-AVC is that *B* slices are coded in a manner in which some macroblocks or blocks may use a weighted average of two distinct motion-compensated prediction values for building the prediction signal. However, as another extension of the corresponding functionality beyond MPEG2 video, this does not imply the restriction to the case of using a superposition of forward and backward prediction signals in the classical sense. In fact, the concept of *generalized B pictures* in H.264/MPEG4-AVC allows any arbitrary pair of reference pictures to be utilized for the prediction of each region (as exemplified in Fig. 4). For that purpose, two distinct ways of indexing the multi-picture buffer are maintained for *B* slices, which are referred to as the first (*"list 0"*) and second (*"list 1"*) reference picture lists, respectively. The ordering of these lists is signaled by the encoder. It is also worth noting that by decoupling the ordering of pictures for display and referencing purposes in H.264/MPEG4-AVC, greater flexibility can be achieved, particularly with respect to the control of the structural decoding delay caused by using reference pictures that are displayed later than other pictures that use them as references in the decoding process. This flexibility has been shown to have increasing importance over time, including its use as a fundamental part of the new SVC and upcoming MVC extensions [7–9].

Depending on which reference picture list is used for forming the prediction signal, three different types of interpicture prediction are distinguished in *B* slices: *list 0*, *list 1*, and *bi-predictive*, where the latter uses a superposition of list 0 and list 1 prediction signals and is the key feature provided by *B* slices. With a similar partitioning as specified for *P* slices, the three different interpicture prediction types in *B* slices can be chosen separately for each macroblock partition or sub-macroblock partition. Additionally, *B*-slice macroblocks or sub-macroblocks can also be coded in so-called direct mode without the need to transmit any additional motion information. If no prediction residual data are transmitted for a direct-coded macroblock, it is also referred to as *skipped*, and skipped macroblocks can be indicated very efficiently as for the skip mode in *P* slices [10].

Transform, Scaling, and Quantization

As already noted above, H.264/MPEG4-AVC also uses transform coding of the prediction residual. However, in contrast to prior video coding standards, such as MPEG2 or H.263, which use a 2D discrete cosine transform (DCT) of size 8 × 8, H.264/MPEG4-AVC specifies a set of integer transforms of different block sizes. In all version 1 related profiles, a 4 × 4 integer transform [5] is applied to both the luma and chroma components of the prediction residual signal. An additional $M \times N$ transform stage is further applied to all resulting DC coefficients in the case of the luma component of a macroblock that is coded using the 16 × 16 intra-coding type (with $N = M = 4$) as well as in the case of both chroma components (with the values of $N, M \in \{2,4\}$ depending on the chroma format). For these additional transform stages, separable



■ **Figure 5.** *CABAC block diagram.*

combinations of the four-tap Hadamard transform and two-tap Haar/Hadamard transform are applied.

Besides the important property of low computational complexity, the use of those small block-size transforms in H.264/MPEG4-AVC has the advantage of significantly reducing ringing artifacts. For high-fidelity video, however, the preservation of smoothness and texture generally benefits from a representation with longer basis functions. A better trade-off between these conflicting objectives can be achieved by making use of the 8 × 8 integer transform specified in the FREXt amendment as an additional transform type for coding the luma residual signal. This 8 × 8 block transform is a close approximation of the 2D 8 × 8 DCT, and provides the benefit of allowing efficient implementations in integer arithmetic [11, 12]. In fact, all integer transforms in H.264/MPEG4-AVC as well as their corresponding inverse transforms can be implemented in a very cost-efficient way since only shift and add operations in $(8 + b)$-bit arithmetic precision are required for processing b-bit input video.

As an additional degree of freedom in the FREXt profiles, the encoder has the choice between using the 4 × 4 or 8 × 8 transform in order to adapt the representation of the luma residual signal to its specific characteristics on a macroblock-by-macroblock basis. This adaptive choice is coupled together with related parts of the decoding process; for example, by disallowing use of the 8 × 8 transform when the prediction block size is smaller than 8 × 8.

For the quantization of transform coefficients, H.264/MPEG4-AVC uses uniform-reconstruction quantizers (URQs). One of 52 quantizer step size scaling factors is selected for each macroblock by a quantization parameter (QP). The scaling operations are arranged so that there is a doubling in quantization step size for each increment of six in the value of QP. The quantized transform coefficients of a block generally are scanned in a zig-zag fashion and further processed using the entropy coding methods described below. In addition to the basic step-size control, the FREXt amendment also supports encoder-specified scaling matrices for a perceptual tuned, frequency-dependent quantization (a capability similar to that found in MPEG2 video).

In-Loop Deblocking Filter

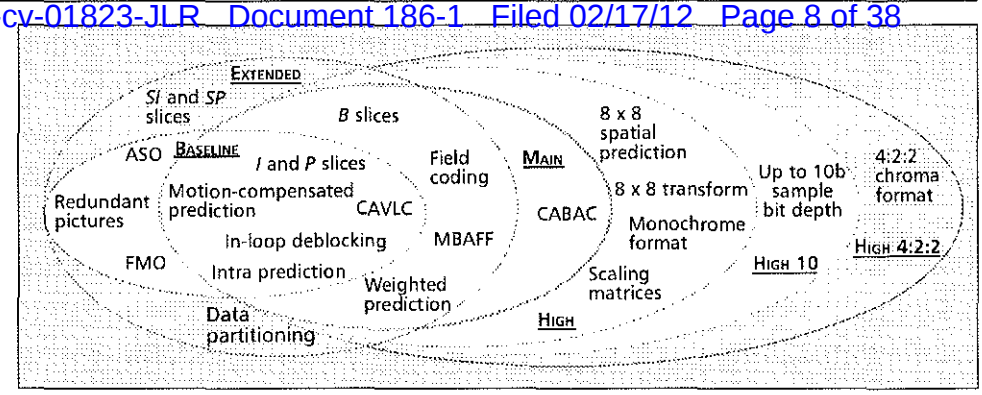
Due to coarse quantization at low bit rates, block-based coding typically results in visually noticeable discontinuities along the block boundaries. If no further provision is made to deal with this, these artificial discontinuities may also diffuse into the interior of blocks by means of the motion-compensated prediction process. The removal of such blocking artifacts can provide a substantial improvement in perceptual quality. For that purpose, H.264/MPEG4-AVC defines a deblocking filter that operates within the predictive coding loop, and thus constitutes a required component of the decoding process. The filtering process exhibits a high degree of content adaptivity on different levels, from the slice level along the edge level down to the level of individual samples. As a result, the blockiness is reduced without much affecting the sharpness of the content. Consequently, the subjective quality is significantly improved. At the same time, the filter reduces bit rate by typically 5–10 percent while producing the same objective quality as the non-filtered video.

Entropy Coding

In H.264/MPEG4-AVC, many syntax elements are coded using the same highly-structured infinite-extent variable-length code (VLC), called a zero-order exponential-Golomb code. A few syntax elements are also coded using simple fixed-length code representations. For the remaining syntax elements, two types of entropy coding are supported.

When using the first entropy-coding configuration, which is intended for lower-complexity (esp. software-based) implementations, the exponential-Golomb code is used for nearly all syntax elements except those of quantized transform coefficients, for which a more sophisticated method called context-adaptive variable length coding (CAVLC) is employed. When using CAVLC, the encoder switches between different

The Baseline profile was targeted at applications in which a minimum of computational complexity and a maximum of error robustness are required, whereas the Main profile was aimed at applications that require a maximum of coding efficiency, with somewhat less emphasis on error robustness.



■ **Figure 6.** *Illustration of H.264/MPEG4-AVC profiles.*

VLC tables for various syntax elements, depending on the values of the previously transmitted syntax elements in the same slice. Since the VLC tables are designed to match the conditional probabilities of the context, the entropy coding performance is improved from that of schemes that do not use context-based adaptivity.

The entropy coding performance is further improved if the second configuration is used, which is referred to as context-based adaptive binary arithmetic coding (CABAC) [5]. As depicted in Fig. 5, the CABAC design is based on three components: binarization, context modeling, and binary arithmetic coding. Binarization enables efficient binary arithmetic coding by mapping nonbinary syntax elements to sequences of bits referred to as *bin strings*. The bins of a bin string can each be processed in either an arithmetic coding mode or a *bypass* mode. The latter is a simplified coding mode that is chosen for selected bins such as sign information or lesser-significance bins in order to speed up the overall decoding (and encoding) processes. The arithmetic coding mode provides the largest compression benefit, where a bin may be context-modeled and subsequently arithmetic encoded. As a design decision, in most cases only the most probable bin of a syntax element is supplied with external context modeling, which is based on previously decoded (encoded) bins. The compression performance of the arithmetic-coded bins is optimized by adaptive estimation of the corresponding (context-conditional) probability distributions. The probability estimation and the actual binary arithmetic coding are conducted using a multiplication-free method that enables efficient implementations in hardware and software. Compared to CAVLC, CABAC can typically provide reductions in bit rate of 10–20 percent for the same objective video quality when coding SDTV/HDTV signals.

PROFILES AND LEVELS

Profiles and levels specify conformance points that provide interoperability between encoder and decoder implementations within applications of the standard and between various applications that have similar functional requirements. A profile defines a set of syntax features for use in generating conforming bitstreams, whereas a level places constraints on certain key parameters of the bitstream such as maximum bit rate and maximum picture size. All decoders conforming to a specific profile and level must support all features included in that profile when constrained as specified for the level. Encoders are not required to make effective use of any particular set of features supported in a profile and level but must not violate the syntax feature set and associated constraints. This implies in particular that conformance to any specific profile and level, although it ensures interoperability with decoders, does not provide any guarantees of end-to-end reproduction quality. Figure 6 illustrates the current six profiles of H.264/MPEG4-AVC and their corresponding main features, as further discussed below.

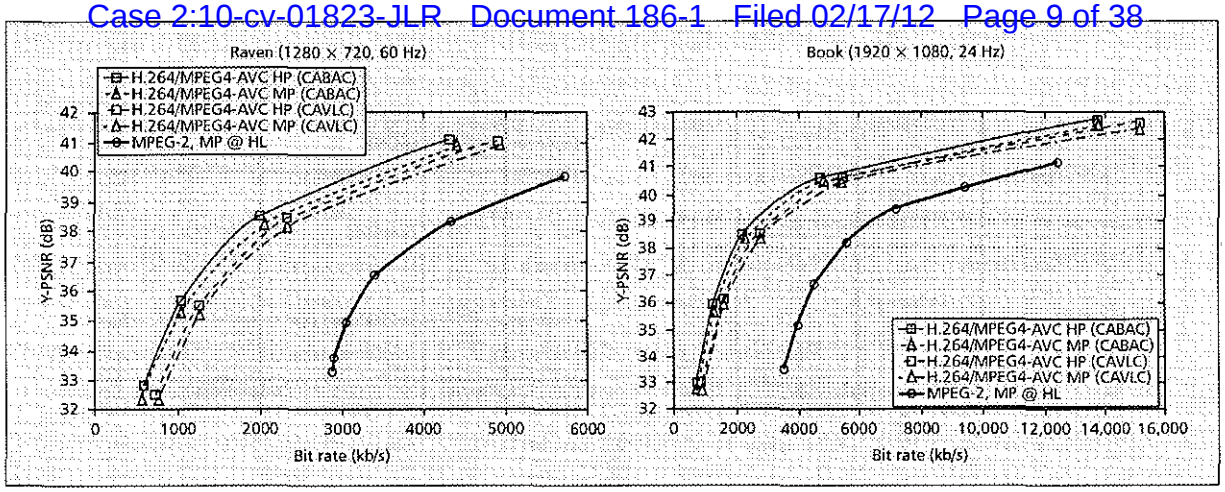
THE BASELINE, MAIN, AND EXTENDED PROFILE (VERSION 1)

In the first version of H.264/MPEG4-AVC three profiles were defined: the Baseline, Extended, and Main profiles. The Baseline profile supports all features in H.264/MPEG4-AVC, v. 1 (2003), except the following three feature sets:

- Set 1: *B* slices, field picture coding, macroblock-adaptive switching between frame and field coding (MBAFF), and weighted prediction
- Set 2: CABAC
- Set 3: *SP* and *SI* slices, and slice data partitioning

The first and second of these three feature sets is supported by the Main profile (MP), in addition to the features supported by the Baseline profile except for the FMO feature and some other enhanced error resilience features [4]. The Extended profile supports all features of the Baseline profile, and the first and third above sets of features, but not CABAC.

Roughly speaking, the Baseline profile was targeted at applications in which a minimum of computational complexity and a maximum of error robustness are required, whereas the Main profile was aimed at applications that require a maximum of coding efficiency, with somewhat less emphasis on error robustness. The Extended profile was designed to provide a compromise between the Baseline and Main profile capabilities with an additional focus on the specific needs



■ **Figure 7.** *Left: Objective performance for the "Raven" sequence (left) and "Book" sequence (right) comparing H.264/MPEG4-AVC HP, MP (both using CABAC and CAVLC), and MPEG2 MP@HL.*

of video streaming applications, and further added robustness to errors and packet losses.

High Profiles in the FRExt Amendment (Version 3)

As depicted in Fig. 6, the H.264/MPEG4-AVC FRExt amendment specifies, in addition to the three profiles of v. 1, a set of three additional profiles constructed as nested sets of capabilities built on top of the Main profile. As their common intersection, the High profile (HP) contains the most relevant FRExt tools for further improving coding efficiency. Relative to the Main profile, these tools imply only a moderate (if any) increase in complexity in terms of both implementation and computational costs (at the decoder side). Therefore, the High profile, with its restriction to 8-bit video in 4:2:0 chroma format, has overtaken the Main profile for prospective applications of H.264/MPEG4-AVC in typical SD and HD consumer applications. Two other profiles, called the *High 10* and *High 4:2:2* profiles, further extend the capability of the standard to include more demanding applications requiring higher sample precision (up to 10 b/sample) and higher chroma formats (up to 4:2:2).[1]

R-D Performance

A comparison of the compression performance achievable with H.264/MPEG4-AVC v. 1 profiles can be found in [5]. Here we also focus on a demonstration of the additional benefit that can be obtained by using some of the HP-specific coding tools. More specifically, we have evaluated the gain in objective performance for the HP-specific 8 × 8 coding tools in terms of objective performance. For that purpose, we have performed a series of coding simulations by using a test set of seven progressive HD sequences with different characteristics and different spatiotemporal resolutions (four 720p sequences with 1280 × 720 @ 60 Hz and three 1080p sequences with 1920 × 1080 @ 24 Hz). The coding simulations were carried out using the H.264/MPEG4-AVC reference software encoder (version JM 9.2) and an MPEG2 Main profile (MP@HL) conforming encoder. To provide a fair comparison, both encoders have been controlled using the same R-D optimized encoding strategy [5]. For both encoders, an *I*-frame refresh was performed every 500 ms, and two non-reference *B* pictures have been inserted between each two successive *P* pictures. Full-search motion estimation was performed with a search range of ±32 integer pixels. For H.264/MPEG4-AVC up to three reference frames were used.

As an example of the outcome of our experiments, Fig. 7 shows the R-D curves for the 720p "Raven" (left) and 1080p "Book" (right) sequences comparing H.264/MPEG4-AVC HP (including 8 × 8 coding tools), MP with both CABAC and CAVLC, and MPEG2. Since these sequences are characterized by predominantly highly textured content, relatively large gains can be obtained in favor of HP due to the better frequency selectivity of the 8 × 8 luma transform. Averaged over the whole HD test set and a quality range of 33–39 dB peak signal-to-noise ratio (PSNR), HP achieves bit rate savings of about 10 percent relative to MP (both using CABAC), as shown in Table 1. If, however, the 8 × 8 tool set of HP is used in conjunction with CAVLC, an average loss of about 18 percent is observed relative to HP using CABAC, which means that the CAVLC-driven HP leads, on average, to objectively lower performance than that measured for the CABAC-driven MP (Table 1).

In comparison with MPEG2, the H.264/MPEG4-AVC High profile coder (with 8 × 8 coding tools and CABAC enabled) achieves average bit rate savings of about 59 percent when measured over the entire test set and investigated PSNR range.

Conclusions

The new H.264/MPEG4-AVC video coding standard was developed and standardized collaboratively by both the ITU — Telecommunication

[1] *Originally, the FRExt amendment included another, so-called high 4:4:4 profile which, at the time of writing this article, is in the process of being removed from the specification, as the JVT plans to replace it with at least two new profiles of a somewhat different design that are yet to be finalized.*

	Average bit rate savings relative to:		
Coder	H.264/MPEG4-AVC HP using CAVLC	H.264/MPEG4-AVC MP using CABAC	MPEG2 MP@HL
H.264/MPEG4-AVC HP using CABAC	17.9%	9.9%	58.8%

■ **Table 1.** *Average bit rate savings for H.264/MPEG4-AVC HP using CABAC entropy coding relative to H.264/MPEG4-AVC HP using CAVLC, H.264/MPEG4-AVC MP using CABAC, and MPEG2 MP@HL.*

Standardization Sector (ITU-T) VCEG and ISO/IEC MPEG organizations. H.264/MPEG4-AVC represents a number of advances in standardized video coding technology, in terms of both coding efficiency enhancement and flexibility for effective use over a broad variety of network types and application domains. Its video coding layer design is based on conventional block-based motion-compensated hybrid video coding concepts, but with some important innovations relative to prior standards. We summarize some of the important differences thusly:

- Enhanced motion-compensated prediction and spatial intra prediction capabilities
- Use of 4 × 4 and 8 × 8 (FREXt only) transforms in integer precision
- Content-adaptive in-loop deblocking filter
- Enhanced entropy coding methods

When used well together, the features of the new design provide significant bit rate savings for equivalent perceptual quality relative to the performance of prior standards. This is especially true for use of the High profile related coding tools.

Acknowledgment

The authors thank the experts of ITU-T VCEG, ISO/IEC MPEG, and the ITU-T/ISO/IEC Joint Video Team for their contributions. The authors would also like to thank the anonymous reviewers for their helpful comments and suggestions.

Further Reading

Further information and documents of the JVT project are available online at http://ftp3.itu.ch/av-arch/jvt-site/. The reader interested in individual technical subjects within the scope of version 1 of H.264/MPEG4-AVC is referred to a special journal issue on H.264/MPEG4-AVC [5]. Additional information about FREXt-specific technical aspects can be found in [13, 14], while [15] includes some background information about the history and development of the new standard as well as information related to the recent deployment and adoption status.

References

[1] ITU-T Rec. H.262 and ISO/IEC 13818-2 (MPEG2), "Generic Coding of Moving Pictures and Associated Audio Information — Part 2: Video," Nov. 1994.
[2] ITU-T Rec. H.263, "Video Coding for Low Bit Rate Communication," v1, Nov. 1995; v2, Jan. 1998; v3, Nov. 2000.
[3] ISO/IEC JTC 1, "Coding of Audio-Visual Objects — Part 2: Visual," ISO/IEC 14496-2 (MPEG4 Visual Version 1), April 1999; Amendment 1 (Version 2), Feb. 2000; Amendment 4 (streaming profile), Jan. 2001.
[4] ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG4-AVC), "Advanced Video Coding for Generic Audiovisual Services," v1, May, 2003; v2, Jan. 2004; v3 (with FREXt), Sept. 2004; v4, July 2005.
[5] A. Luthra, G. J. Sullivan, and T. Wiegand, Eds., Special issue on the "H.264/AVC Video Coding Standard," *IEEE Trans. Circuits and Sys. for Video Tech.*, vol. 13, no. 7, July 2003.
[6] G. J. Sullivan *et al.*, "Draft text of H.264/AVC Fidelity Range Extensions Amendment," ISO/IEC MPEG and ITU-T VCEG, JVT-L047, Redmond, WA, July 2004.
[7] T. Wiegand *et al.*, "Joint Draft 5: Scalable Video Coding," ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-R201, Bangkok, Thailand, Jan. 2006.
[8] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the Scalable H.264/MPEG4-AVC Extension," to be presented, *IEEE Int'l. Conf. Image Processing*, Atlanta, GA, Oct. 2006.
[9] K. Müller *et al.*, "Multi-View Video Coding Based on H.264/AVC Using Hierarchical B-Frames," *Proc. PCS 2006*, Beijing, China, Apr. 2006.
[10] A. M. Tourapis *et al.*, "Direct Mode Coding for Bipredictive Slices in the H.264 Standard," *IEEE Trans. Circuits and Sys. for Video Tech.*, vol. 15, no. 1, Jan. 2005, pp. 119–26.
[11] S. Gordon, D. Marpe, and T. Wiegand, "Simplified Use of 8x8 Transforms," ISO/IEC MPEG and ITU-T VCEG, JVT-K028, Munich, Germany, Mar. 2004.
[12] F. Bossen, "ABT Cleanup and Complexity Reduction," ISO/IEC MPEG and ITU-T VCEG, JVT-E087, Geneva, Switzerland, Oct. 2002.
[13] D. Marpe, T. Wiegand, and S. Gordon, "H.264/MPEG4-AVC Fidelity Range Extensions: Tools, Profiles, Performance, and Application Areas," *IEEE Int'l. Conf. Image Processing*, vol. 1, Sept. 2005, pp. 593–96.
[14] G. J. Sullivan, P. Topiwala, and A. Luthra, "The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions," *SPIE Annual Conf. Apps. of Digital Image Processing XXVII*, Special Session on Advances in the New Emerging Standard H.264/AVC, Aug., 2004, pp. 454–74.
[15] G. J. Sullivan, "The H.264/MPEG4-AVC Video Coding Standard and Its Deployment Status," *SPIE Conf. Visual Commun. and Image Processing*, Beijing, China, July 2005.

Biographies

Detlev Marpe [M'00] received a Dr.-Ing. degree from the University of Rostock, Germany, in 2005, and a Dipl.-Math. degree (with highest honors) from the Technical University of Berlin (TUB), Germany, in 1990. From 1991 to 1993 he was a research and teaching assistant in the Department of Mathematics at TUB. Since 1994 he has been involved in several industrial and research projects in the area of still image coding, image processing, video coding, and video streaming. In 1999 he joined the Fraunhofer Institute for Telecommunications — Heinrich-Hertz-Institute (HHI), Berlin, Germany, where as a project manager in the Image Processing Department he is currently responsible for research projects focused on the development of advanced video coding and video transmission technologies. He has published more than 40 journal and conference articles in the area of image and video processing, and holds several international patents in this field. He has been involved in ITU-T and ISO/IEC standardization activities for still image and video coding, to which he has contributed about 50 input documents. From 2001 to 2003, as an Ad Hoc Group Chairman in the Joint Video Team of ITU-T VCEG and ISO/IEC MPEG, he was responsible for the development of the CABAC entropy coding scheme within the H.264/MPEG4-AVC standardization project. He also served as co-editor of the H.264/MPEG4-AVC FREXt Amendment in 2004. As a co-founder of daViKo GmbH, a Berlin-based startup company involved in the development of serverless multipoint videoconferencing products for intranet or Internet collaboration, he received the Prime Prize of the 2001 Multimedia Startup Competition founded by the German Federal Ministry of Economics and Technology. In 2004 he received the Fraunhofer Prize for outstanding scientific achievements in solving application related problems and the ITG Award of the German Society for Information Technology. His current research interests include still image and video coding, image and video communication, as well as computer vision and information theory.

Thomas Wiegand [M'05] is head of the Image Communication Group in the Image Processing Department of the Fraunhofer Institute for Telecommunications — HHI. He received a Dipl.-Ing. degree in electrical engineering from the Technical University of Hamburg-Harburg, Germany, in 1995 and a Dr.-Ing. degree from the University of Erlan-

gen-Nuremberg, Germany, in 2000. From 1993 to 1994 he was a visiting researcher at Kobe University, Japan. In 1995 he was a visiting scholar at the University of California at Santa Barbara, where he started his research on video compression and transmission. Since then he has published several conference and journal papers on the subject and has contributed successfully to the ITU-T Video Coding Experts Group (ITU-T SG16 Q.6 — VCEG)/ISO/IEC JTC1/SC29/WG11 — MPEG) Joint Video Team (JVT) standardization efforts and holds various international patents in this field. From 1997 to 1998 he was a visiting researcher at Stanford University, California. In October 2000 he was appointed Associate Rapporteur of ITU-T VCEG. In December 2001 he was appointed Associated Rapporteur/Co-Chair of the JVT. In February 2002 he was appointed editor of the H.264/AVC video coding standard. In January 2005 he was appointed Associate Chair of MPEG Video. In 1998 he received the SPIE VCIP Best Student Paper Award. In 2004 he received the Fraunhofer Prize for outstanding scientific achievements in solving application related problems and the ITG Award of the German Society for Information Technology. Since January 2006 he is an Associate Editor of *IEEE Transactions on Circuits and Systems for Video Technology*. His research interests include image and video compression, communication and signal processing, as well as vision and computer graphics.

GARY J. SULLIVAN (S'83-M'91-SM'01-F'06) received B.S. and M.Eng. degrees in electrical engineering from the University of Louisville J.B. Speed School of Engineering,Kentucky, in 1982 and 1983, respectively. He received Ph.D. and Engineer degrees in electrical engineering from the University of California, Los Angeles, in 1991. He is the ITU-T rapporteur/chairman of the ITU-T Video Coding Experts Group (VCEG), a co-chairman of the ISO/IEC Moving Picture Experts Group (MPEG), and a co-chairman of the Joint Video Team (JVT), which is a joint project between the VCEG and MPEG organizations. He has led ITU-T VCEG (ITU-T Q.6/SG16) since 1996 and is also the ITU-T video liaison representative to MPEG. In MPEG (ISO/IEC JTC1/SC29/WG11), in addition to his current service as a co-chair of its video work, he also served as the chairman of MPEG video from March 2001 to May 2002. In the JVT he was the JVT chairman for the development of the next-generation H.264/MPEG4-AVC video coding standard and its fidelity-range extensions (FREXt), and is now its co-chairman for the development of the scalable video coding (SVC) extensions. He received the Technical Achievement award of the International Committee on Technology Standards (INCITS) in 2005 for his work on H.264/MPEG4-AVC and other video standardization topics. He holds the position of video architect in the Core Media Processing Team in the Windows Digital Media division of Microsoft Corporation. At Microsoft he also designed and remains lead engineer for the DirectX® Video Acceleration API/DDI video decoding feature of the Microsoft Windows® operating system platform. Prior to joining Microsoft in 1999, he was the manager of communications core research at PictureTel Corporation, the quondam world leader in videoconferencing communication. He was previously a Howard Hughes Fellow and member of technical staff in the Advanced Systems Division of Hughes Aircraft Corporation, and a terrain-following radar system software engineer for Texas Instruments. His research interests and areas of publication include image and video compression, rate-distortion optimization, motion estimation and compensation, scalar and vector quantization, and error-/packet-loss-resilient video coding.

EXHIBIT AC

INTERNATIONAL TELECOMMUNICATION UNION

ITU-T H.261

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(03/93)

{This document has included corrections to typographical errors listed in Annex 5 to COM 15R 16-E dated June 1994. - Sakae OKUBO}

LINE TRANSMISSION OF NON-TELEPHONE SIGNALS

VIDEO CODEC FOR AUDIOVISUAL SERVICES AT *p* × 64 kbit/s

ITU-T Recommendation H.261

(Previously "CCITT Recommendation")

FOREWORD

The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of the International Telecommunication Union. The ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Conference (WTSC), which meets every four years, established the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

ITU-T Recommendation H.261 was revised by the ITU-T Study Group XV (1988-1993) and was approved by the WTSC (Helsinki, March 1-12, 1993).

NOTES

1 As a consequence of a reform process within the International Telecommunication Union (ITU), the CCITT ceased to exist as of 28 February 1993. In its place, the ITU Telecommunication Standardization Sector (ITU-T) was created as of 1 March 1993. Similarly, in this reform process, the CCIR and the IFRB have been replaced by the Radiocommunication Sector.

In order not to delay publication of this Recommendation, no change has been made in the text to references containing the acronyms "CCITT, CCIR or IFRB" or their associated entities such as Plenary Assembly, Secretariat, etc. Future editions of this Recommendation will contain the proper terminology related to the new ITU structure.

2 In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

CONTENTS

Page

1 Scope 1

2 Brief specification 1

2.1 Video input and output 2
2.2 Digital output and input 2
2.3 Sampling frequency 2
2.4 Source coding algorithm 2
2.5 Bit rate 2
2.6 Symmetry of transmission 3
2.7 Error handling 3
2.8 Multipoint operation 3

3 Source coder 3

3.1 Source format 3
3.2 Video source coding algorithm 3
3.3 Coding control 6
3.4 Forced updating 6

4 Video multiplex coder 7

4.1 Data structure 7
4.2 Video multiplex arrangement 7
4.3 Multipoint considerations 18

5 Transmission coder 19

5.1 Bit rate 19
5.2 Video data buffering 19
5.3 Video coding delay 20
5.4 Forward error correction for coded video signal 20

Annex A – Inverse transform accuracy specification 21

Annex B – Hypothetical reference decoder 22

Annex C – Codec delay measurement method 23

Annex D – Still image transmission 24

2.6 Symmetry of transmission

The codec may be used for bidirectional or unidirectional visual communication.

2.7 Error handling

The transmitted bit-stream contains a BCH code (Bose, Chaudhuri and Hocquengham) (511,493) forward error correction code. Use of this by the decoder is optional.

2.8 Multipoint operation

Features necessary to support switched multipoint operation are included.

3 Source coder

3.1 Source format

The source coder operates on non-interlaced pictures occurring 30 000/1001 (approximately 29.97) times per second. The tolerance on picture frequency is ± 50 ppm.

Pictures are coded as luminance and two colour difference components (Y, C_B and C_R). These components and the codes representing their sampled values are as defined in CCIR Recommendation 601.

Black = 16

White = 235

Zero colour difference = 128

Peak colour difference = 16 and 240.

These values are nominal ones and the coding algorithm functions with input values of 1 through to 254.

Two picture scanning formats are specified.

In the first format (CIF), the luminance sampling structure is 352 pels per line, 288 lines per picture in an orthogonal arrangement. Sampling of each of the two colour difference components is at 176 pels per line, 144 lines per picture, orthogonal. Colour difference samples are sited such that their block boundaries coincide with luminance block boundaries as shown in Figure 2. The picture area covered by these numbers of pels and lines has an aspect ratio of 4:3 and corresponds to the active portion of the local standard video input.

NOTE – The number of pels per line is compatible with sampling the active portions of the luminance and colour difference signals from 525- or 625-line sources at 6.75 and 3.375 MHz, respectively. These frequencies have a simple relationship to those in CCIR Recommendation 601.

The second format, quarter-CIF (QCIF), has half the number of pels and half the number of lines stated above. All codecs must be able to operate using QCIF. Some codecs can also operate with CIF.

Means shall be provided to restrict the maximum picture rate of encoders by having at least 0, 1, 2 or 3 non-transmitted pictures between transmitted ones. Selection of this minimum number and CIF or QCIF shall be by external means (for example via Recommendation H.221).

3.2 Video source coding algorithm

The source coder is shown in generalized form in Figure 3. The main elements are prediction, block transformation and quantization.

The prediction error (INTER mode) or the input picture (INTRA mode) is subdivided into 8 pel by 8 line blocks which are segmented as transmitted or non-transmitted. Further, four luminance blocks and the two spatially corresponding colour difference blocks are combined to form a macroblock as shown in Figure 10.

4.2.2.1 Group of blocks start code (GBSC) (16 bits)

A word of 16 bits, 0000 0000 0000 0001.

4.2.2.2 Group number (GN) (4 bits)

Four bits indicating the position of the group of blocks. The bits are the binary representation of the number in Figure 6. Group numbers 13, 14 and 15 are reserved for future use. Group number 0 is used in the PSC.

4.2.2.3 Quantizer information (GQUANT) (5 bits)

A fixed length codeword of 5 bits which indicates the quantizer to be used in the group of blocks until overridden by any subsequent MQUANT. The codewords are the natural binary representations of the values of QUANT (see 4.2.4) which, being half the step sizes, range from 1 to 31.

4.2.2.4 Extra insertion information (GEI) (1 bit)

A bit which when set to "1" signals the presence of the following optional data field.

4.2.2.5 Spare information (GSPARE) (0/8/16 . . . bits)

If GEI is set to "1", then 9 bits follow consisting of 8 bits of data (GSPARE) and then another GEI bit to indicate if a further 9 bits follow and so on. Encoders must not insert GSPARE until specified by the CCITT. Decoders must be designed to discard GSPARE if GEI is set to 1. This will allow the CCITT to specify future "backward" compatible additions in GSPARE.

NOTE – Emulation of start codes may occur if the future specification of GSPARE has no restrictions on the final GSPARE data bits.

4.2.3 Macroblock layer

Each GOB is divided into 33 macroblocks as shown in Figure 8. A macroblock relates to 16 pels by 16 lines of Y and the spatially corresponding 8 pels by 8 lines of each of C_B and C_R.

Data for a macroblock consists of an MB header followed by data for blocks (see Figure 9). MQUANT, MVD and CBP are present when indicated by MTYPE.

1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22
23	24	25	26	27	28	29	30	31	32	33

FIGURE 8/H.261

Arrangement of macroblocks in a GOB

MBA	MTYPE	MQUANT	MVD	CBP	Block data

FIGURE 9/H.261

Structure of macroblock layer

TABLE 3/H.261

VLC table for MVD

MVD	Code
–16 & 16	0000 0011 001
–15 & 17	0000 0011 011
–14 & 18	0000 0011 101
–13 & 19	0000 0011 111
–12 & 20	0000 0100 001
–11 & 21	0000 0100 011
–10 & 22	0000 0100 11
–9 & 23	0000 0101 01
–8 & 24	0000 0101 11
–7 & 25	0000 0111
–6 & 26	0000 1001
–5 & 27	0000 1011
–4 & 28	0000 111
–3 & 29	0001 1
–2 & 30	0011
–1	011
0	1
1	010
2 & –30	0010
3 & –29	0001 0
4 & –28	0000 110
5 & –27	0000 1010
6 & –26	0000 1000
7 & –25	0000 0110
8 & –24	0000 0101 10
9 & –23	0000 0101 00
10 & –22	0000 0100 10
11 & –21	0000 0100 010
12 & –20	0000 0100 000
13 & –19	0000 0011 110
14 & –18	0000 0011 100
15 & –17	0000 0011 010

4.2.4 Block layer

A macroblock comprises four luminance blocks and one of each of the two colour difference blocks (see Figure 10).

Data for a block consists of codewords for transform coefficients followed by an end of block marker (see Figure 11). The order of block transmission is as in Figure 10.

4.2.4.1 Transform coefficients (TCOEFF)

Transform coefficient data is always present for all six blocks in a macroblock when MTYPE indicates INTRA. In other cases MTYPE and CBP signal which blocks have coefficient data transmitted for them. The quantized transform coefficients are sequentially transmitted according to the sequence given in Figure 12.

The most commonly occurring combinations of successive zeros (RUN) and the following value (LEVEL) are encoded with variable length codes. Other combinations of (RUN, LEVEL) are encoded with a 20-bit word consisting of 6 bits ESCAPE, 6 bits RUN and 8 bits LEVEL. For the variable length encoding there are two code tables, one being used for the first transmitted LEVEL in INTER, INTER+MC and INTER+MC+FIL blocks, the second for all other LEVELs except the first one in INTRA blocks which is fixed length coded with 8 bits.

TABLE 4/H.261

VLC table for CBP

CBP	Code	CBP	Code
60	111	35	0001 1100
4	1101	13	0001 1011
8	1100	49	0001 1010
16	1011	21	0001 1001
32	1010	41	0001 1000
12	1001 1	14	0001 0111
48	1001 0	50	0001 0110
20	1000 1	22	0001 0101
40	1000 0	42	0001 0100
28	0111 1	15	0001 0011
44	0111 0	51	0001 0010
52	0110 1	23	0001 0001
56	0110 0	43	0001 0000
1	0101 1	25	0000 1111
61	0101 0	37	0000 1110
2	0100 1	26	0000 1101
62	0100 0	38	0000 1100
24	0011 11	29	0000 1011
36	0011 10	45	0000 1010
3	0011 01	53	0000 1001
63	0011 00	57	0000 1000
5	0010 111	30	0000 0111
9	0010 110	46	0000 0110
17	0010 101	54	0000 0101
33	0010 100	58	0000 0100
6	0010 011	31	0000 0011 1
10	0010 010	47	0000 0011 0
18	0010 001	55	0000 0010 1
34	0010 000	59	0000 0010 0
7	0001 1111	27	0000 0001 1
11	0001 1110	39	0000 0001 0
19	0001 1101		

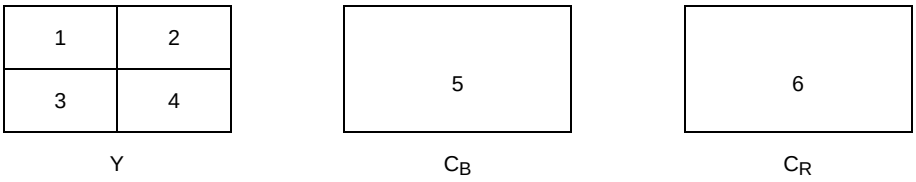


FIGURE 10/H.261

Arrangement of blocks in a macroblock

EXHIBIT AD

INTERNATIONAL TELECOMMUNICATION UNION

ITU-T H.262

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(07/95)

TRANSMISSION OF NON-TELEPHONE SIGNALS

INFORMATION TECHNOLOGY – GENERIC CODING OF MOVING PICTURES AND ASSOCIATED AUDIO INFORMATION: VIDEO

ITU-T Recommendation H.262

(Previously "CCITT Recommendation")

FOREWORD

ITU (International Telecommunication Union) is the United Nations Specialized Agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of the ITU. Some 179 member countries, 84 telecom operating entities, 145 scientific and industrial organizations and 38 international organizations participate in ITU-T which is the body which sets world telecommunications standards (Recommendations).

The approval of Recommendations by the Members of ITU-T is covered by the procedure laid down in WTSC Resolution No. 1 (Helsinki, 1993). In addition, the World Telecommunication Standardization Conference (WTSC), which meets every four years, approves Recommendations submitted to it and establishes the study programme for the following period.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC. The text of ITU-T Recommendation H.262 was approved on 10th of July 1995. The identical text is also published as ISO/IEC International Standard 13818-2.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

CONTENTS

		Page
Summary		iii
Introduction		iv
Intro. 1	Purpose	iv
Intro. 2	Application	iv
Intro. 3	Profiles and levels	iv
Intro. 4	The scalable and the non-scalable syntax	v
1	Scope	1
2	Normative references	1
3	Definitions	2
4	Abbreviations and symbols	8
4.1	Arithmetic operators	8
4.2	Logical operators	8
4.3	Relational operators	9
4.4	Bitwise operators	9
4.5	Assignment	9
4.6	Mnemonics	9
4.7	Constants	9
5	Conventions	9
5.1	Method of describing bitstream syntax	9
5.2	Definition of functions	10
5.3	Reserved, forbidden and marker_bit	11
5.4	Arithmetic precision	11
6	Video bitstream syntax and semantics	11
6.1	Structure of coded video data	11
6.2	Video bitstream syntax	23
6.3	Video bitstream semantics	39
7	The video decoding process	62
7.1	Higher syntactic structures	63
7.2	Variable length decoding	63
7.3	Inverse scan	66
7.4	Inverse quantisation	68
7.5	Inverse DCT	72
7.6	Motion compensation	73
7.7	Spatial scalability	87
7.8	SNR scalability	100
7.9	Temporal scalability	105
7.10	Data partitioning	108
7.11	Hybrid scalability	109
7.12	Output of the decoding process	110
8	Profiles and levels	114
8.1	ISO/IEC 11172-2 compatibility	115
8.2	Relationship between defined profiles	115
8.3	Relationship between defined levels	117
8.4	Scalable layers	118
8.5	Parameter values for defined profiles, levels and layers	121

		Page
Annex A – Discrete cosine transform		125
Annex B – Variable length code tables		126
B.1	Macroblock addressing	126
B.2	Macroblock type	127
B.3	Macroblock pattern	132
B.4	Motion vectors	133
B.5	DCT coefficients	134
Annex C – Video buffering verifier		143
Annex D – Features supported by the algorithm		148
D.1	Overview	148
D.2	Video formats	148
D.3	Picture quality	149
D.4	Data rate control	149
D.5	Low delay mode	150
D.6	Random access/channel hopping	150
D.7	Scalability	150
D.8	Compatibility	157
D.9	Differences between this Specification and ISO/IEC 11172-2	157
D.10	Complexity	160
D.11	Editing encoded bitstreams	160
D.12	Trick modes	160
D.13	Error resilience	161
D.14	Concatenated sequences	168
Annex E – Profile and level restrictions		169
E.1	Syntax element restrictions in profiles	169
E.2	Permissible layer combinations	180
Annex F – Bibliography		201

Summary

This Recommendation | International Standard specifies coded representation of video data and the decoding process required to reconstruct pictures. It provides a generic video coding scheme which serves a wide range of applications, bit rates, picture resolutions and qualities. Its basic coding algorithm is a hybrid of motion compensated prediction and DCT. Pictures to be coded can be either interlaced or progressive. Necessary algorithmic elements are integrated into a single syntax, and a limited number of subsets are defined in terms of Profile (functionalities) and Level (parameters) to facilitate practical use of this generic video coding standard.

3.74 interlace: The property of conventional television frames where alternating lines of the frame represent different instances in time. In an interlaced frame, one of the field is meant to be displayed first. This field is called the first field. The first field can be the top field or the bottom field of the frame.

3.75 I-field picture: A field structure I-Picture.

3.76 I-frame picture: A frame structure I-Picture.

3.77 I-picture; intra-coded picture: A picture coded using information only from itself.

3.78 intra coding: Coding of a macroblock or picture that uses information only from that macroblock or picture.

3.79 level: A defined set of constraints on the values which may be taken by the parameters of ITU-T Rec. H.262 | ISO/IEC 13818-2 within a particular profile. A profile may contain one or more levels. In a different context, level is the absolute value of a non-zero coefficient (see "run").

3.80 layer: In a scalable hierarchy denotes one out of the ordered set of bitstreams and (the result of) its associated decoding process (implicitly including decoding of **all** layers below this layer).

3.81 layer bitstream: A single bitstream associated to a specific layer (always used in conjunction with layer qualifiers, e. g. "enhancement layer bitstream").

3.82 lower layer: A relative reference to the layer immediately below a given enhancement layer (implicitly including decoding of **all** layers below this enhancement layer).

3.83 luminance component: A matrix, block or single sample representing a monochrome representation of the signal and related to the primary colours in the manner defined in the bitstream. The symbol used for luminance is Y.

3.84 Mbit: 1 000 000 bits.

3.85 macroblock: The four 8 by 8 blocks of luminance data and the two (for 4:2:0 chrominance format), four (for 4:2:2 chrominance format) or eight (for 4:4:4 chrominance format) corresponding 8 by 8 blocks of chrominance data coming from a 16 by 16 section of the luminance component of the picture. Macroblock is sometimes used to refer to the sample data and sometimes to the coded representation of the sample values and other data elements defined in the macroblock header of the syntax defined in ITU-T Rec. H.262 | ISO/IEC 13818-2. The usage is clear from the context.

3.86 motion compensation: The use of motion vectors to improve the efficiency of the prediction of sample values. The prediction uses motion vectors to provide offsets into the past and/or future reference frames or reference fields containing previously decoded sample values that are used to form the prediction error.

3.87 motion estimation: The process of estimating motion vectors during the encoding process.

3.88 motion vector: A two-dimensional vector used for motion compensation that provides an offset from the coordinate position in the current picture or field to the coordinates in a reference frame or reference field.

3.89 non-intra coding: Coding of a macroblock or picture that uses information both from itself and from macroblocks and pictures occurring at other times.

3.90 opposite parity: The opposite parity of top is bottom, and vice versa.

3.91 P-field picture: A field structure P-Picture.

3.92 P-frame picture: A frame structure P-Picture.

3.93 P-picture; predictive-coded picture: A picture that is coded using motion compensated prediction from past reference fields or frame.

3.94 parameter: A variable within the syntax of ITU-T Rec. H.262 | ISO/IEC 13818-2 which may take one of a range of values. A variable which can take one of only two values is called a flag.

3.95 parity (of field): The parity of a field can be top or bottom.

3.96 past reference frame (field): A past reference frame (field) is a reference frame (field) that occurs at an earlier time than the current picture in display order.

6.1.2.2 Restricted slice structure

In certain defined levels of defined profiles a restricted slice structure illustrated in Figure 6-9 shall be used. In this case every macroblock in the picture shall be enclosed in a slice.

Where a defined level of a defined profile requires that the slice structure obeys the restrictions detailed in this subclause, the term "restricted slice structure" may be used.

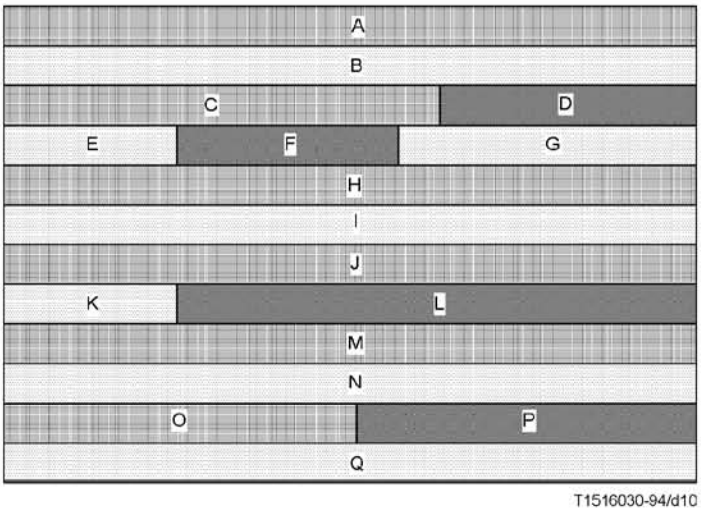


Figure 6-9 – Restricted slice structure

6.1.3 Macroblock

A **macroblock** contains a section of the luminance component and the spatially corresponding chrominance components. The term macroblock can either refer to source and decoded data or to the corresponding coded data elements. A skipped macroblock is one for which no information is transmitted (see 7.6.6). There are three chrominance formats for a macroblock, namely, 4:2:0, 4:2:2 and 4:4:4 formats. The orders of blocks in a macroblock shall be different for each different chrominance format and are illustrated below.

A 4:2:0 Macroblock consists of 6 blocks. This structure holds 4 Y, 1 Cb and 1 Cr Blocks and the block order is depicted in Figure 6-10.

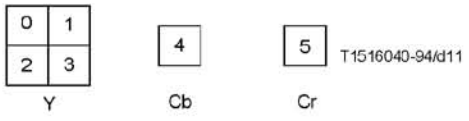


Figure 6-10 – 4:2:0 Macroblock structure

A 4:2:2 Macroblock consists of 8 blocks. This structure holds 4 Y, 2 Cb and 2 Cr Blocks and the block order is depicted in Figure 6-11.

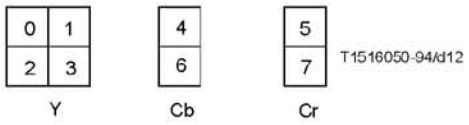


Figure 6-11 – 4:2:2 Macroblock structure

A 4:4:4 Macroblock consists of 12 blocks. This structure holds 4 Y, 4 Cb and 4 Cr Blocks and the block order is depicted in Figure 6-12.

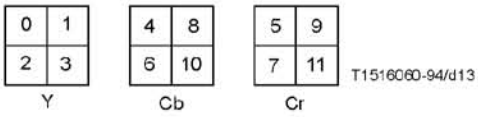


Figure 6-12 – 4:4:4 Macroblock structure

In frame pictures, where both frame and field DCT coding may be used, the internal organisation within the macroblock is different in each case.

- In the case of frame DCT coding, each block shall be composed of lines from the two fields alternately. This is illustrated in Figure 6-13.
- In the case of field DCT coding, each block shall be composed of lines from only one of the two fields. This is illustrated in Figure 6-14.

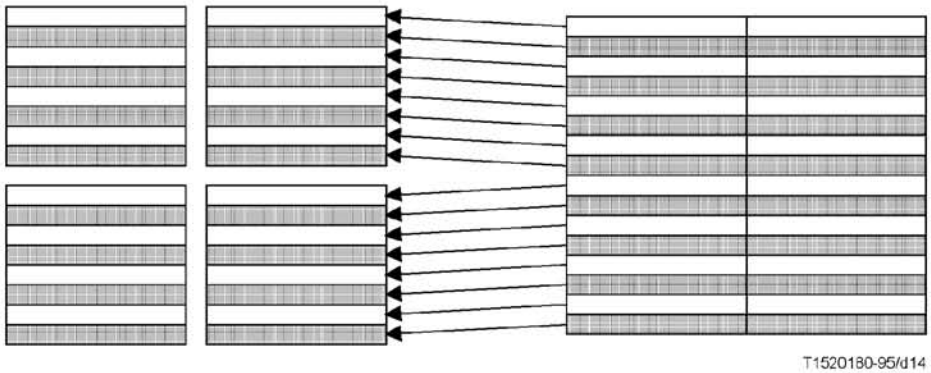


Figure 6-13 – Luminance macroblock structure in frame DCT coding

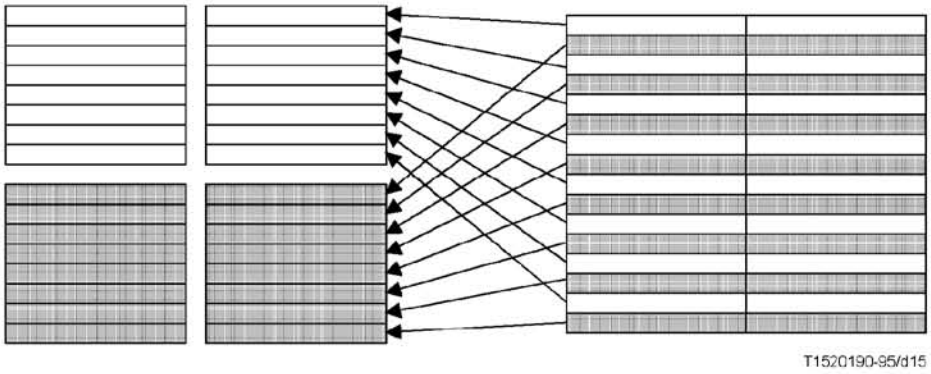


Figure 6-14 – Luminance macroblock structure in field DCT coding

In the case of chrominance blocks the structure depends upon the chrominance format that is being used. In the case of 4:2:2 and 4:4:4 formats (where there are two blocks in the vertical dimension of the macroblock) the chrominance blocks are treated in exactly the same manner as the luminance blocks. However, in the 4:2:0 format the chrominance blocks shall always be organised in frame structure for the purposes of DCT coding. It should, however, be noted that field based predictions may be made for these blocks which will, in the general case, require that predictions for 8×4 regions (after half-sample filtering) must be made.

In field pictures, each picture only contains lines from one of the fields. In this case each block consists of lines taken from successive lines in the picture as illustrated by Figure 6-13.

6.1.4 Block

The term "**block**" can refer either to source and reconstructed data or to the DCT coefficients or to the corresponding coded data elements.

When "block" refers to source and reconstructed data it refers to an orthogonal section of a luminance or chrominance component with the same number of lines and samples. There are 8 lines and 8 samples in the block.

6.2 Video bitstream syntax

6.2.1 Start codes

Start codes are specific bit patterns that do not otherwise occur in the video stream.

Each start code consists of a start code prefix followed by a start code value. The start code prefix is a string of twenty three bits with the value zero followed by a single bit with the value one. The start code prefix is thus the bit string '0000 0000 0000 0000 0000 0001'.

The start code value is an eight bit integer which identifies the type of start code. Most types of start code have just one start code value. However, slice_start_code is represented by many start code values, in this case the start code value is the slice_vertical_position for the slice.

All start codes shall be byte aligned. This shall be achieved by inserting bits with the value zero before the start code prefix such that the first bit of the start code prefix is the first (most significant) bit of a byte.

Table 6-1 defines the slice start code values for the start codes used in the video bitstream.

The use of the start codes is defined in the following syntax description with the exception of the sequence_error_code. The sequence_error_code has been allocated for use by a media interface to indicate where uncorrectable errors have been detected.

EXHIBIT AE

INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11 N2202

Tokyo, March 1998

INFORMATION TECHNOLOGY -

CODING OF AUDIO-VISUAL OBJECTS: VISUAL

ISO/IEC 14496-2

Committee Draft

Draft of 28 May, 1998

Contents

1. Introduction ..... **vii**
1.1 Purpose vii
1.2 Application vii
1.3 Profiles and levels vii
1.4 Object based coding syntax viii
1.4.1 Video object viii
1.4.2 Face object viii
1.4.3 Mesh object ix
1.4.4 Overview of the object based nonscalable syntax ix
1.4.5 Generalized scalability xi
1.5 Error Resilience xii

1. Scope **13**

2. Normative references **13**

3. Definitions **15**

4. Abbreviations and symbols **23**
4.1 Arithmetic operators 23
4.2 Logical operators 24
4.3 Relational operators 24
4.4 Bitwise operators 24
4.5 Conditional operators 24
4.6 Assignment 24
4.7 Mnemonics 24
4.8 Constants 25

5. Conventions **26**
5.1 Method of describing bitstream syntax 26
5.2 Definition of functions 27
5.2.1 Definition of bytealigned() function 27
5.2.2 Definition of nextbits_bytealigned() function 27
5.2.3 Definition of next_start_code() function 27
5.2.4 Definition of next_resync_marker() function 27
5.2.5 Definition of transparent_mb() function 28
5.2.6 Definition of transparent_block() function 28
5.3 Reserved, forbidden and marker_bit 28
5.4 Arithmetic precision 28

6. Visual bitstream syntax and semantics **29**
6.1 Structure of coded visual data 29
6.1.1 Visual object sequence 29
6.1.2 Visual object 30
6.1.3 Video object 30
6.1.4 Mesh object 36
6.1.5 Face object 38
6.2 Visual bitstream syntax 41
6.2.1 Start codes 41
6.2.2 Visual Object Sequence and Visual Object 42
6.2.3 Video Object 44
6.2.4 Video Object Layer 45
6.2.5 Group of Video Object Plane 49
6.2.6 Video Object Plane and Video Plane with Short Header 50
6.2.7 Macroblock 63
6.2.8 Block 68
6.2.9 Still Texture Object 70
6.2.10 Mesh Object 79
6.2.11 Face Object 81
6.3 Visual bitstream semantics 92
6.3.1 Semantic rules for higher syntactic structures 92
6.3.2 Visual Object Sequence and Visual Object 92
6.3.3 Video Object 98
6.3.4 Video Object Layer 98
6.3.5 Group of Video Object Plane 105
6.3.6 Video Object Plane and Video Plane with Short Header 106
6.3.7 Macroblock related 117
6.3.8 Block related 120

6.3.9 Still texture object 120
6.3.10 Mesh related 125
6.3.11 Face object 127

7. The visual decoding process 134

7.1 Video decoding process 134
7.2 Higher syntactic structures 135
7.3 Texture decoding 136
7.3.1 Variable length decoding 136
7.3.2 Inverse scan 138
7.3.3 Intra dc and ac prediction for intra macroblocks 139
7.3.4 Inverse quantisation 141
7.3.5 Inverse DCT 144
7.4 Shape decoding 144
7.4.1 Higher syntactic structures 145
7.4.2 Macroblock decoding 145
7.4.3 Arithmetic decoding 155
7.4.4 Grayscale Shape Decoding 157
7.5 Motion compensation decoding 159
7.5.1 Padding process 160
7.5.2 Half sample interpolation 163
7.5.3 General motion vector decoding process 164
7.5.4 Unrestricted motion compensation 166
7.5.5 Vector decoding processing and motion-compensation in progressive P-VOP 166
7.5.6 Overlapped motion compensation 168
7.5.7 Temporal prediction structure 170
7.5.8 Vector decoding process of non-scalable progressive B-VOPs 171
7.5.9 Motion compensation in non-scalable progressive B-VOPs 171
7.6 Interlaced video decoding 175
7.6.1 Field DCT and DC and AC Prediction 175
7.6.2 Motion compensation 175
7.7 Sprite decoding 184
7.7.1 Higher syntactic structures 185
7.7.2 Sprite Reconstruction 185
7.7.3 Low-latency sprite reconstruction 186
7.7.4 Sprite reference point decoding 187
7.7.5 Warping 188
7.7.6 Sample reconstruction 189
7.7.7 Scalable sprite decoding 190
7.8 Generalized scalable decoding 191
7.8.1 Temporal scalability 193
7.8.2 Spatial scalability 196
7.9 Still texture object decoding 200
7.9.1 Decoding of the DC subband 200
7.9.2 ZeroTree Decoding of the Higher Bands 201
7.9.3 Inverse Quantization 207
7.10 Mesh object decoding 211
7.10.1 Mesh geometry decoding 212
7.10.2 Decoding of mesh motion vectors 215
7.11 Face object decoding 218
7.11.1 Frame based face object decoding 218
7.11.2 DCT based face object decoding 218
7.11.3 Decoding of the viseme parameter fap 1 220

7.11.4 Decoding of the viseme parameter fap 2 220
7.11.5 Fap masking 221
7.12 Output of the decoding process 222
7.12.1 Video data 222
7.12.2 2D Mesh data 222
7.12.3 Face animation parameter data 222

8. Visual-Systems Composition Issues 223
8.1 Temporal Scalability Composition 223
8.2 Sprite Composition 224

9. Profiles and Levels 225
9.1 Visual Object Profiles 225
9.2 Visual Combination Profiles 227
9.3 Visual Combination Profiles@Levels 227
9.3.1 Natural Visual 227
9.3.2 Synthetic Visual 227
2.4.3 Synthetic/Natural Hybrid Visual 228

10. Annex A 229
10.1 Discrete cosine transform for video texture 229
10.2 Discrete wavelet transform for still texture 230
10.2.1 Adding the mean 230
10.2.2 wavelet filter 230
10.2.3 Symmetric extension 231
10.2.4 Decomposition level 231
10.2.5 Shape adaptive wavelet filtering and symmetric extension 231

11. Annex B 233
11.1 Variable length codes 233
11.1.1 Macroblock type 233
11.1.2 Macroblock pattern 234
11.1.3 Motion vector 237
11.1.4 DCT coefficients 239
11.1.5 Shape Coding 249
11.1.6 Sprite Coding 254
11.1.7 DCT based facial object decoding 255
11.2 Arithmetic Decoding 265
11.2.1 Aritmetic decoding for still texture object 265
11.2.2 Arithmetic decoding for shape decoding 269
11.2.3 Face Object Decoding 271

12. Annex C 273

13. Annex D 284

14. Annex E 285
14.1 Error resilience 285
14.1.1 Resynchronization 285
14.1.2 Data Partitioning 286
14.1.3 Reversible VLC 287
14.1.4 Decoder Operation 287
14.1.5 Adaptive Intra Refresh (AIR) Method 291
14.2 Complexity Estimation 294
14.2.1 Video Object Layer Class 294

14.2.2 Video Object Plane Class 297
14.2.3 Video Object Plane 297
14.2.4 Resynchronization in Case of Unknown Video Header Format 299

15. Annex F 301

15.1 Segmentation for VOP Generation 301
15.1.1 Introduction 301
15.1.2 Description of a combined temporal and spatial segmentation framework 301
15.1.3 References 303
15.2 Bounding Rectangle of VOP Formation 305
15.3 Postprocessing for Coding Noise Reduction 306
15.3.1 Deblocking filter 306
15.3.2 Deringing filter 307
15.3.3 Further issues 309

16. Annex G 310

17. Annex H 311

18. Annex I 312

19. Annex J 313

20. Annex K 314

20.1 Introduction 314
20.2 Decoding Process of a View-Dependent Object 314
20.2.1 General Decoding Scheme 314
20.2.2 Computation of the View-Dependent Scalability parameters 316
20.2.3 VD mask computation 318
20.2.4 Differential mask computation 319
20.2.5 DCT coefficients decoding 319
20.2.6 Texture update 319
20.2.7 IDCT 320

21. Annex L 321

21.1 Introduction 321
21.2 Description of the set up of a visual decoder (informative) 321
21.2.1 Processing of decoder configuration information 322
21.3 Specification of decoder configuration information 323
21.3.1 VideoObject 323
21.3.2 StillTextureObject 323
21.3.3 MeshObject 324
21.3.4 FaceObject 324

22. Annex M 325

3.68 **I-VOP; intra-coded VOP**: A VOP coded using information only from itself.

3.69 **intra coding**: Coding of a macroblock or VOP that uses information only from that macroblock or VOP.

3.70 **intra shape coding**: Shape coding that does not use any temporal prediction.

3.71 **inter shape coding**: Shape coding that uses temporal prediction.

3.72 **level**: A defined set of constraints on the values which may be taken by the parameters of this specification within a particular profile. A profile may contain one or more levels. In a different context, level is the absolute value of a non-zero coefficient (see "run").

3.73 **layer**: In a scalable hierarchy denotes one out of the ordered set of bitstreams and (the result of) its associated decoding process.

3.74 **layered bitstream**: A single bitstream associated to a specific layer (always used in conjunction with layer qualifiers, e. g. "enhancement layer bitstream")

3.75 **lower layer**: A relative reference to the layer immediately below a given enhancement layer (implicitly including decoding of *all* layers below this enhancement layer)

3.76 **luminance component**: A matrix, block or single sample representing a monochrome representation of the signal and related to the primary colours in the manner defined in the bitstream. The symbol used for luminance is Y.

3.77 **Mbit**: 1 000 000 bits

3.78 **macroblock**: The four 8×8 blocks of luminance data and the two (for 4:2:0 chrominance format) corresponding 8×8 blocks of chrominance data coming from a 16×16 section of the luminance component of the picture. Macroblock is sometimes used to refer to the sample data and sometimes to the coded representation of the sample values and other data elements defined in the macroblock header of the syntax defined in this part of this specification. The usage is clear from the context.

3.79 **mesh**: A 2D triangular mesh refers to a planar graph which tessellates a video object plane into triangular patches. The vertices of the triangular mesh elements are referred to as node points. The straight-line segments between node points are referred to as edges. Two triangles are adjacent if they share a common edge.

3.80 **mesh geometry**: The spatial locations of the node points and the triangular structure of a mesh.

3.81 **mesh motion**: The temporal displacements of the node points of a mesh from one time instance to the next.

3.82 **motion compensation**: The use of motion vectors to improve the efficiency of the prediction of sample values. The prediction uses motion vectors to provide offsets into the past and/or future reference VOPs containing previously decoded sample values that are used to form the prediction error.

3.83 **motion estimation**: The process of estimating motion vectors during the encoding process.

The following is an example of VOPs taken from the beginning of a video object layer. In this example there are two coded B-VOPs between successive coded P-VOPs and also two coded B-VOPs between successive coded I- and P-VOPs. VOP '1I' is used to form a prediction for VOP '4P'. VOPs '4P' and '1I' are both used to form predictions for VOPs '2B' and '3B'. Therefore the order of coded VOPs in the coded sequence shall be '1I', '4P', '2B', '3B'. However, the decoder shall display them in the order '1I', '2B', '3B', '4P'.

At the encoder input,

1	2	3	4	5	6	7	8	9	10	11	12	13
I	B	B	P	B	B	P	B	B	I	B	B	P

At the encoder output, in the coded bitstream, and at the decoder input,

1	4	2	3	7	5	6	10	8	9	13	11	12
I	P	B	B	P	B	B	I	B	B	P	B	B

At the decoder output,

1	2	3	4	5	6	7	8	9	10	11	12	13

6.1.3.8 Macroblock

A macroblock contains a section of the luminance component and the spatially corresponding chrominance components. The term macroblock can either refer to source and decoded data or to the corresponding coded data elements. A skipped macroblock is one for which no information is transmitted. Presently there is only one chrominance format for a macroblock, namely, 4:2:0 format. The orders of blocks in a macroblock is illustrated below:

A 4:2:0 Macroblock consists of 6 blocks. This structure holds 4 Y, 1 Cb and 1 Cr Blocks and the block order is depicted in Figure 6-5.

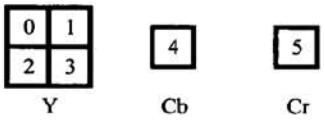


Figure 6-5 -- 4:2:0 Macroblock structure

The organisation of VOPs into macroblocks is as follows.

For the case of a progressive VOP, the interlaced flag (in the VOP header) is set to "0" and the organisation of lines of luminance VOP into macroblocks is called frame organization and is illustrated in Figure 6.4. In this case, frame DCT coding is employed.

For the case of interlaced VOP, the interlaced flag is set to "1" and the organisation of lines of luminance VOP into macroblocks can be either frame organization or field organization and thus both frame and field DCT coding may be used in the VOP.

- In the case of frame DCT coding, each luminance block shall be composed of lines from two fields alternately. This is illustrated in Figure 6-9.
- In the case of field DCT coding, each luminance block shall be composed of lines from only one of the two fields. This is illustrated in Figure 6-10.

Only frame DCT coding is applied to the chrominance blocks. It should be noted that field based predicitons may be applied for these chrominance blocks which will require predictions of 8x4 regions (after half-sample filtering).

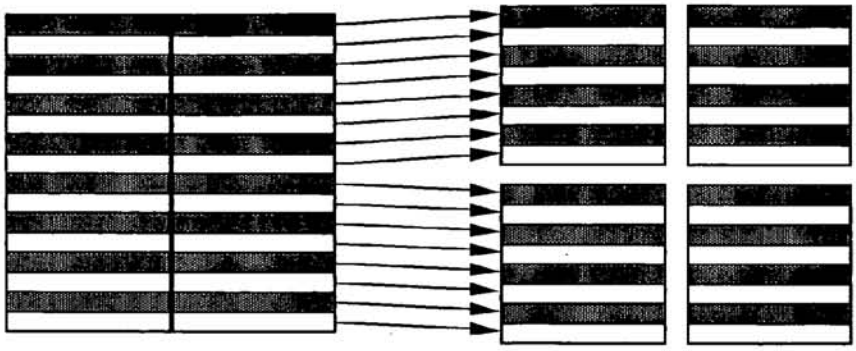


Figure 6-6 -- Luminance macroblock structure in field DCT coding

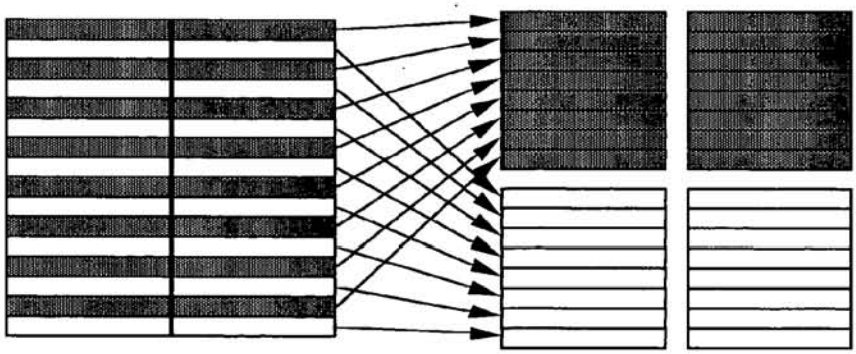


Figure 6-7 -- Luminance macroblock structure in field DCT coding

6.1.3.9 Block

The term **block** can refer either to source and reconstructed data or to the DCT coefficients or to the corresponding coded data elements.

When the block refers to source and reconstructed data it refers to an orthogonal section of a luminance or chrominance component with the same number of lines and samples. There are 8 lines and 8 samples/line in the block.

6.1.4 Mesh object

A 2D triangular *mesh* refers to a tessellation of a 2D visual object plane into triangular patches. The vertices of the triangular patches are called *node points*. The straight-line segments between node points are called *edges*. Two triangles are *adjacent* if they share a common edge.